Virtual Reality Engineering Education

By: Quan Nghiem | Advisor: Dr. Erika Parsons | Sponsors: Matthew Fuentes & Dr. Chris Byrne

Overview

Problem:

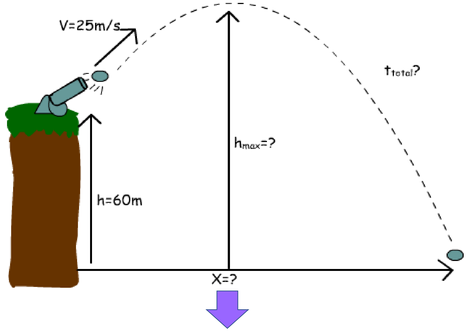
- Engineering students have trouble visualizing problems

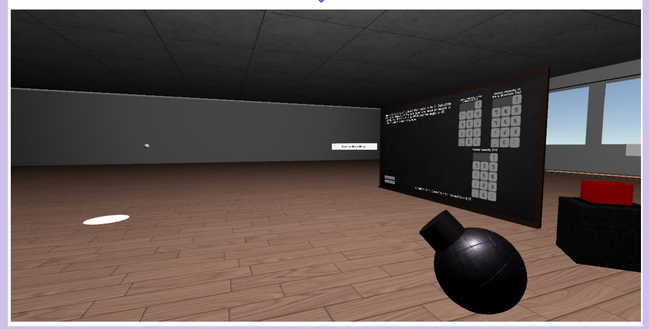
Goal:

- Develop a VR application for physics simulation
- Develop a physics library to work with Unity physics engine for better simulation
- Develop a framework for building engineering problems and demos
- Provide a modularized project for future undergraduate research

Why VR:

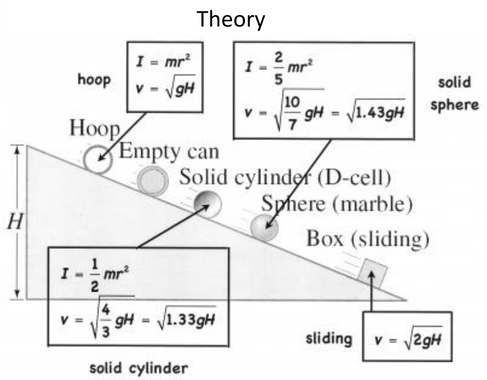
- Remove the need to build or purchase physical models for demo
- Better intractability and immersion for students



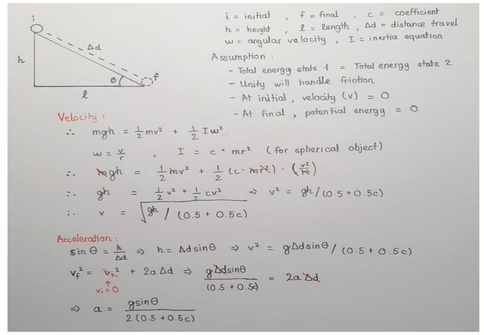


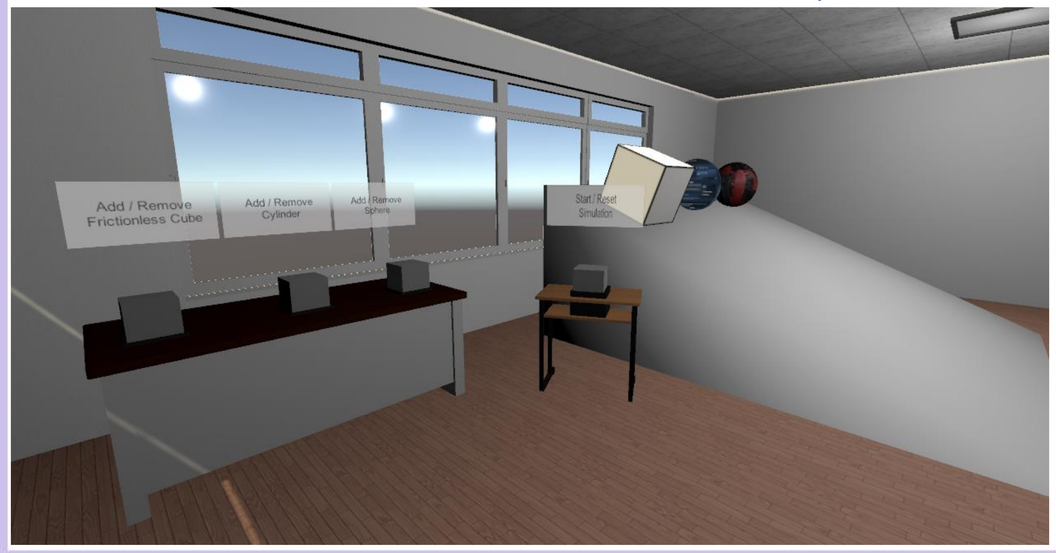
Development Process

Theory



Problem Solving





Physic Library Script

- Contains functions to calculate physics equations
- Implements in conjunction with Unity built-in physic engine for more realistic simulation
- Improve control over angular velocity and acceleration mechanism

```
private Vector3 calculateTrajectoryVelocity(Vector3 target, Vector3 origin)
{
    // Define gravity
    float gravity = Math.Abs(Physics.gravity.y);

    // Define the distance x and y
    Vector3 distance = target - origin;
    Vector3 horizontalDistance = distance;
    horizontalDistance.y = 0f;

    // Actual length of the distance
    float Dy = distance.y;
    float Dxz = horizontalDistance.magnitude;

    float height = (Dxz * Mathf.Tan(angle * Mathf.Deg2Rad)) / 4;

    Vector3 Vy = Vector3.up * Mathf.Sqrt(2 * gravity * height);
    float time = Mathf.Sqrt(2 * height / gravity) + Mathf.Sqrt(-2 * (Dy - height) / gravity);
    Vector3 Vxz = horizontalDistance / time;

    Vector3 result = Vxz + Vy;
```

Challenges

- Converting from 2D space to 3D space was hard
- Unity basic functions don't work as intended
- Unity doesn't support for translational or rotational kinetic equations
- Few documentation on Unity physics engine inner working
- Unable to meet in person
- Difficult physics concepts
- Update scripts to match the improved user input system

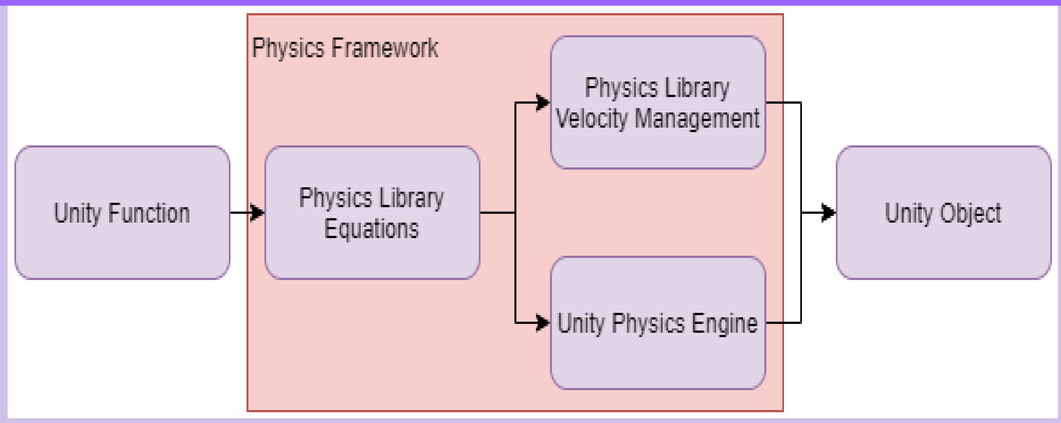
Lessons Learned

- Derive physics formulas before implementing in Unity
- Plan all functionality and features before coding
- Centralize functions to avoid confusing script referencing
- Usability should be done as soon as possible to avoid unnoticed bugs and problems
- How to convert 2D vector to 3D vector
- Static and dynamic physics concepts

Contribution

- Developed the user input keypad system
- Developed and integrated physics framework with Unity physics engine
- Developed part of the environment for the demo

Functionality Overview



Tools Used

unity, zoom

Acknowledgment

Special thank to my advisor, Dr. Erika Parsons, my sponsors, Professor Matthew Fuentes and Dr. Chris Byrne for assisting me in developing the physics engine and retaught some fundamental physics concepts to me.